Rule-oriented method for parameterized computer-aided design

Anne Verroust, François Schonek, Dieter Roller

Rule-oriented method for parameterized computer-aided design

Anne Verroust, François Schonek and Dieter Roller *
LIENS, Ecole Normale Supérieure
CNRS URA 1327
45, rue d'Ulm 75230 PARIS CEDEX 05 FRANCE

Abstract

The paper presents an implementation for computer-aided design with dimensional parameters. The approach is based on the use of an expert system to uncouple constraint equations, and to find a possible sequence for the computation of the geometric elements for given dimension values. A set of rules for the expert system is described that solves the problem for 2D designs. The method is illustrated on an example design.

Keywords. Dimensioning, geometric constraints, CAD system, parameterized designs, geometric-reasoning method.

1 Introduction

Contemporary CAD systems have proven to be effective for generating technical drawings and modeling 3D objects. However, in the conception stage of the design process, most CAD systems still do not have all the required flexibility. One particularly important aspect is the design with dimensional parameters.

Variable dimensions provide means for the designer to create his or her initial design without taking care of the exact dimensions, which in early design stages are unknown anyway. Moreover, in the design to manufacturing process a number of iteration cycles is usually required, before the design meets all its functional and manufacturing related requirements. In many of these cycles, the design can undergo dimensional changes.

Work has been carried out to improve the designer interface in this way.
In [1], the different approaches are classified into three families:

- **primary approaches** that provide a solution in specific cases, like Fitzgerald [2] and Gossard, Zuffante and Sakurai [3] or approaches that introduce an aid through macros, like Cugini, Devoti and Galli [4].
- **algebraic approaches** [5, 6, 7, 8, 9, 10] that transform the dimensioning problem into a numerical problem : the resolution of a system of (non-linear) equations. They use a classic method as Newton-Raphson or an improvement of it to solve

0* The third author's mailing address is: Universität Stuttgart, Institut für Informatik, Breitwiesenstrase 20-22 7000, Stuttgart 80, F.R.G.

the system. However these approaches have limited capabilities in respect to the handling of incompletely specified drawings. Furthermore non-consistent constraining schemes are not rapidly detected (*cf.* [11] for a preprocessing method solving the problem).

- **artificial intelligence oriented approaches** that use inference to construct the drawing of a design progressively. The first approaches [12, 13] use simple rules to fix the design gradually but do not detect the inconsistencies in the constraining scheme. More recently Brüderlin [14], Aldefeld [15] and Sunde [16, 17] have proposed rule oriented approaches which detect and provide an explanation when a part of the drawing is overdetermined and are able to give all the numerical solutions corresponding to the set of constraints.

Our method follows Aldefeld and Sunde's most recent work but our goal is a little different: we focus on the main rules used to evaluate two-dimensional models and on the scope of the rule-based approaches.
This paper is structured as follows. The next section precisely states the problem that is to be handled. Then the main rules used in the expert system, i.e. the creation, construction and verification rules, are explained.

The domain of application of the method is then studied: the class of constraint-based geometric models that can be evaluated by our system is described.

A description of the implementation follows. The characteristics of the expert system shell used are given, together with the different facts used by the rules as well as a list of some rules that are being used by the system.

Finally, the operation of the system is explained on an example.

2 Rule oriented 2D approach

Geometric models that represent 2D computer aided designs are considered. These models consist of oriented line segments with their endpoints and of geometric constraints, such as the following:

- two line segments share a common endpoint
- two points are a given distance apart
- two line segments make a given angle.

In fact, a mechanical design also includes circles and axes of symmetry.

1. Tangency constraints between circles or between a line and a circle, as well as constraints on the radius can be expressed as follows (*cf.* Figure 1):

   ◦ A tangency constraint between a line and a circle is expressed by a right angle constraint between the radius (to the circle-line tangency point) of the circle and the line,

   ◦ A tangency constraint between two circles is expressed by a flat angle constraint between the radii (to the circle-circle tangency point) of the circles,

   ◦ The diameter is translated in a distance constraint of the radius of the circle.

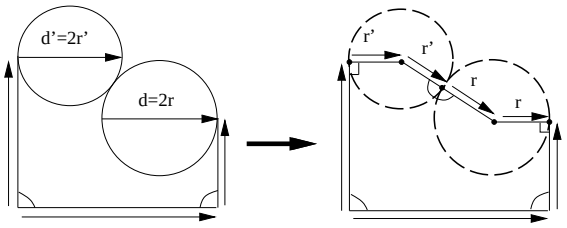

Figure 1: Constraints on circles translated into angle and distance constraints

2. In presence of an axis of symmetry, a distance or an angular constraint in one side of the design induces a similar constraint on the other side. This can be done in a pre-processing stage.

Therefore, it is assumed in this paper that the translations from "circle constraints" and "distance constraint perpendicular to a segment" to "distance and angle constraints" and the insertion of distance or angular constraints due to the presence of axes of symmetry has been already performed.
The problems that are to be solved are as follows:

• For a given model and given values for distance and angle constraints, the coordinates for all the model points have to be evaluated with respect to an original figure, to solve ambiguities during the computation,

• During the creation of a model, overconstrained situations have to be detected as soon as a redundant constraint is inserted.

The method is based on an expert system shell, i.e. the constraints and the points are facts. Rules about these facts are used to evaluate the location of the model points and to detect inconsistent constraining schemes. In fact, to build a real interactive environment one can follow Roller [18], Sunde [16] or Aldefeld [15]:

- permanent evaluation of the model throughout the session and maintenance of the design history for the following reasons:
  - to give immediate explanations to the user when a part of the design is overconstrained,
  - to maintain a part of the construction when a constraint is edited,
  - to handle automatically families of designs that have the same constraining scheme but different numerical values.
- proposal of different solutions to the user when constraints lead to multiple choices.

We have focused on the geometric problem, i.e. on the rules of the expert system rather than providing all interactive tools for the design to the user: these tools can be easily added to our system following Roller, Sunde and Aldefeld.

3 Rules

3.1 Creation rules

We will follow Sunde [16] and use the notions of constrained angle sets and constrained distance sets (in short, CA and CD sets) to build the rules. Let us state in more detail what these two sets are.

- A CA set is a set of pairs of points which corresponding oriented segments are mutually constrained in angle.
- A CD set is a set of points with mutually constrained distances. A frame of reference is attached to each CD set and the location of the points belonging to the CD set are expressed in it. A model is completely constrained when all the points belong to the same CD set.

With these CD sets, the solution can be built using intermediate frames (which seems not to be the case in Aldefeld's method, considering his examples). This distinction is of importance when studying the scope of the method, as seen below.

Definition. In this paper, two CD sets are said to be *adjacent*, if they have one point in common. A segment is said to *belong to a CA set* if its endpoints form a pair of this CA set.

Elementary CA and CD sets are created when constraints are added to the model, as illustrated in Figure 2:

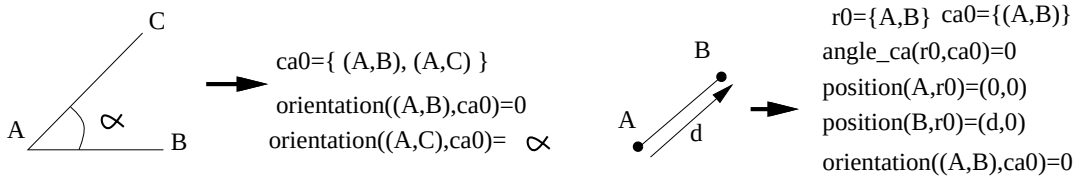

Figure 2: Creation of CA and CD sets

- When an angle constraint α between two directed segments is placed, a CA set is created and the orientations are noted.
- When a distance constraint d between two points is given, a CD and a CA sets are created. The positions of the points in the associated frame reference are $(0, 0)$ and $(d, 0)$. The angle of the CD set with respect to the CA set is noted as 0.

These elementary CA and CD sets are merged using the construction rules.

3.2 Construction rules

With the notion of CA and CD sets, Sunde also gives two rules which result in an enlargement of CA and CD sets and a reduction of the number of these sets:

Rule S1: when a segment belongs to two different CA sets, then these two sets are combined into one CA set.

Rule S2: if two CD sets contain a common point and the angle between them is constrained, then the two CD sets are combined into one CD set.

Apart from these rules and what we be called the "parallelogram" rule, our main construction rules are based on the special cases of the triangle and on one case of quadrilateral. Every time one of these rules is triggered, several CD sets are merged. This is explained in more detail in the next subsections.

3.2.1 Triangle rules

There are three special cases for a triangle:

1. *Triangle specified by three distance constraints: three CD sets defining a triangle.* The triangle rule **T1** computes the intersection of two circles and the three CD sets are merged (*cf.* case T1 of Figure 3)[1].

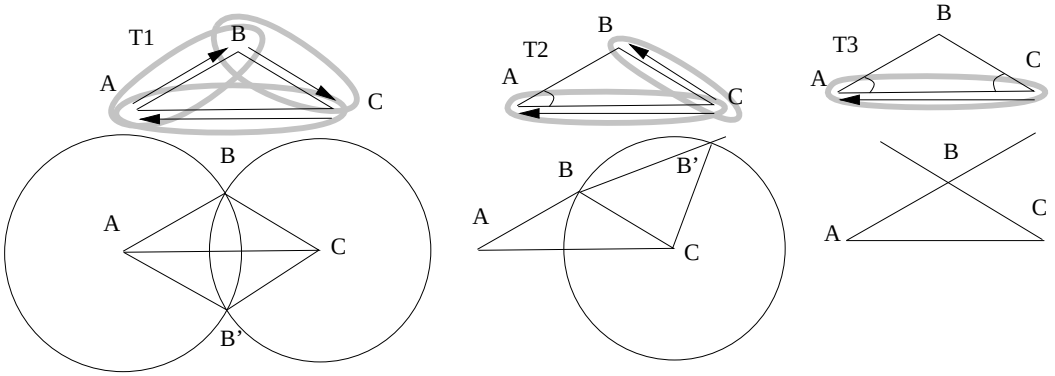

Figure 3: The triangle rules **T1**, **T2** and **T3**

2. *Triangle specified by two distances and an angle not constraining the two CD sets.* The triangle rule **T2** is activated, computing the intersection of a circle and a line and the Two CD sets are joined together (*cf.* case T2 of Figure 3).

3. *Triangle specified by one distance and two angles.* A point belongs to two different lines fixed in a CD set. The triangle rule **T3** computes the intersection of these lines, adding the intersection point to the CD set (*cf.* case T3 of Figure 3).

The computation of the intersection of two circles or of a circle and a line may lead to zero, one, two or, when the circles are identical, an infinite number of solutions. In the first and the last case, there is a numerical impossibility. This is detected during the triggering of the rules. When there are two solutions, one of them is chosen, using an angular criterion on an original figure (*cf.* [19] for details).

[1]In the following Figures, angle constraints between adjacent segment are marked by a curve, distance constraints by an arrow and CD sets by a closed thick curve around the respective points.

3.2.2 Parallelogram rule

Another rule is introduced to manage angle constraints between non adjacent segments as they may appear in technical drawings[2]. This rule assembles non adjacent CD sets constrained by an angle. In this rule, a parallelogram is inserted in the model, by the addition of a point and two segments.

Let us explain how it works on case A of Figure 4 [3].

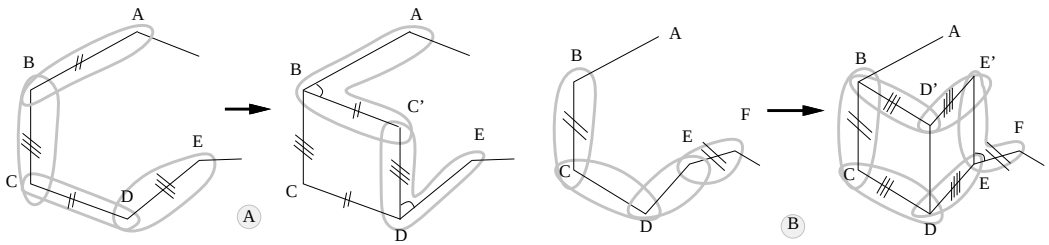

Figure 4: The parallelogram rule.

In this example the segments BC and DE have fixed directions and their lengths are known. It is also the case for AB and CD. Then the point C' is added, with:

- BC' parallel to CD and equal in length
- C'D parallel to BC and equal in length.

The CD sets containing B, C' and A, B can be merged using rule **S2**. The same is true for the CD sets containing D, C' and D, E. Now the problem can be solved, replacing C by C' without complicating the model. In fact, when the positions of B, C' and D are found, the CD sets corresponding to C, D and B, C are combined using rule S2 and the value of C is obtained.

There is a second version of the parallelogram rule involving two parallelograms. It is shown in case B of Figure 4.

3.2.3 Quadrilateral rule

Using the triangle rules and the parallelogram rules, a quadrilateral can be constructed in most cases. One just adds rule **Q** for a quadrilateral in one case: when a quadrilateral is fixed in angle and when two opposite segments are constrained in distance. The problem is reduced to a computation of the intersection of two different lines, as shown in Figure 5.

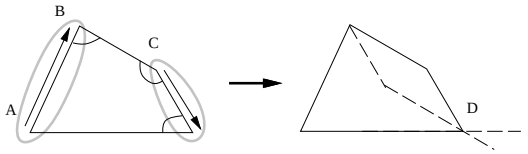

Figure 5: Quadrilateral rule **Q**

We have chosen to introduce the rule **Q** rather than another parallelogram rule when

[2]This powerful rule is not mentioned in Brüderlin, Aldefeld and Sunde's papers [14, 15, 16, 17].

[3]The segments marked with the same number of strokes are fixed together in direction.

two CD sets are in translation along two different lines to avoid unnecessary adjunctions of parallelograms during the computation.

Section 4 shows that the set of construction rules solves the problem for a large class of geometric models.

3.3 Verification rules

The evaluation of a model under a given set of constraints may fail for two different reasons:

- There is a numerical impossibility.

  If a constraint value leads to a numerical impossibility, i.e. the numerical values given for the constraint cannot lead to a solution, then the triangle rules and the quadrilateral rule will detect it. The procedure associated with each rule checks whether the computation is possible or not before any further action is started.

- A part of the model is overdetermined.

  When a new constraint is introduced by the user, all the applicable rules are activated. Then, if the model is computable by our method, the CA sets at each step contain exactly the endpoints of the segments fixed in direction and the CD sets contain the points fixed in distance. Thus, adding a redundant constraint to an already constrained part of the model leads to one of the following cases:

  ◦ The two segments constrained in angle already belong to the same CA set.

  ◦ The two points constrained in distance already belong to the same CD set.

  These conditions are easy to express as rules in the expert system. For each insertion of a constraint, these rules are activated before the construction rules: if a new constraint is redundant, the system refuses to insert it and sends a warning signal to the user.

Note that, when detecting redundancy, it is presumed that all the information have been deduced from the constraints already given by the user. When the model cannot be computed by our set of rules, we cannot ensure that all the overdeterminations have been detected. The same applies in Aldefeld's and Sunde's methods. Thus it is important to characterize the set of two-dimensional models our method can compute.

4 Model coverage

For the two-dimensional problem, it is known that in order to fix n points together, $2n-3$ distance or angle constraints are necessary, if an origin and a rotation about the origin are given[4]. The problem considered here is as follows.

Can any model that is completely constrained by angle and distance constraints be computed by our method, or, if this is not the case, is there a describable subset of models with that property ?

[4]Two unknowns are associated with each point and a distance or an angle constraint gives one equation. As an origin and a rotation are given, three unknowns are fixed. Thus $2n-3$ equations are required to fix the remaining unknowns.

We will see in section 4.3 that there exist models that cannot be computed by our method. We restrict our goal to find a description for a sufficiently large class of models that is covered by our method.
Let us first distinguish between two classes of models consisting of n constrained points.

Definition 1 *Given a model F that includes n points and* $2n-3$ *constraints, the graph* G_F *= (N, E) is defined as follows. N is the set of points of F. E is the set of segments of F upon which there is an angle or a distance constraint.*
The model is "simple" if and only if there exist a simple cycle G' *covering* G_F*. It will be called "non-simple" in the other cases (see Figure 6 for examples).*

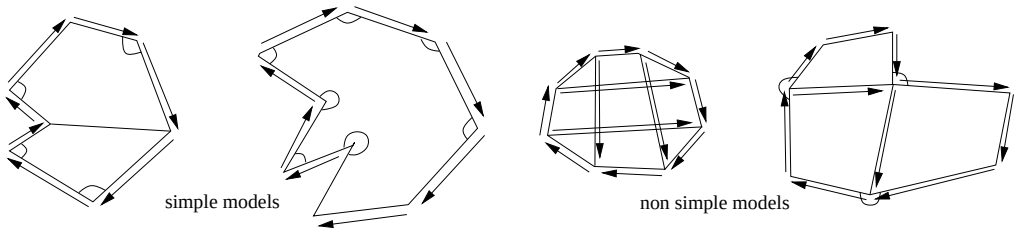

Figure 6: Simple and non simple models.

Proposition 1 *If the model is simple, completely constrained and if the numerical values are consistent, the algorithm computes an evaluated version satisfying the constraints.*

The proof of this proposition is the subject of the next section.

4.1 Case of simple models

Consider the cycle G' covering the graph G_F. All the distance constraints concern adjacent points and angle constraints concern existing edges. G' contains exactly n edges and n points. We need $2n-3$ constraints to properly define the model. This set of constraints contains at most n distance constraints and $n-1$ angle constraints. Thus, three cases are possible.

4.1.1 Simple models constrained by $n-1$ angles and $n-2$ distances

If the model is constrained by $n-1$ angles, the segments are all fixed in direction. Thus, they belong to the same CA set. The distance constraints may either concern

- *One set of adjacent segments*: there is only one CD set for the whole model, containing $n-1$ points as in Figure 7A, then rule **T3** fixes all the points,
- *Two sets of adjacent segments*: there are two CD sets containing the n points as in Figure 7B, rule **Q** computes their union.

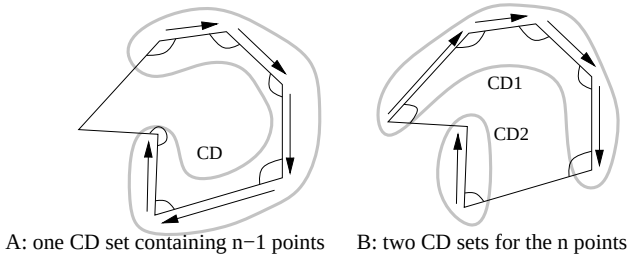

Figure 7: $n-1$ angles and $n-2$ distances: two cases.

4.1.2 Simple models constrained by $n-2$ angles and $n-1$ distances

As there are only $n-2$ angle constraints, two cases are possible, as follows:

- the $n-1$ adjacent segments are fixed in their relative direction (they belong to the same CA set). Let s be the segment not constrained by an angle and s' the segment not constrained by a distance.

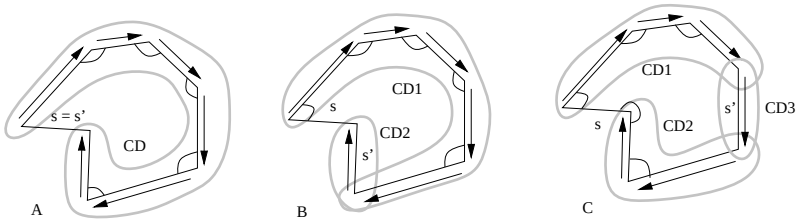

Figure 8: $n-1$ angles constrained together.

– If s = s', all the points belong to the same CD set and the whole model can be computed (Figure 8A).

– If s and s' are adjacent, the $n-1$ points incident to the remaining segments belong to the same CD set. The last point is computed using rule **T2** (Figure 8B).

– If s and s' are not adjacent, the n points are partitioned into two CD sets. A parallelogram rule reduces this case to the previous one.

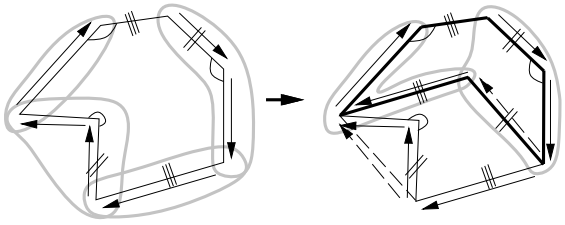

Figure 9: The transformation from a disconnected CA set into a connected one.

- The $n-1$ segments are partitioned into two CA sets. If all the segments belonging to a same CA set are connected, there are two adjacent CD sets whose angle is fixed by a line. Rule **T2** yields the result.

In the other case, a sequence of applications of the "parallelogram" rule connects the segments belonging to a common CA set (*cf.* Figure 9). There is then a model that is similar to the previous one.

4.1.3 Simple models constrained by $n-3$ angles and n distances

The distances of the n segments are fixed and there are at most three CA sets. Two cases are possible, as follows:

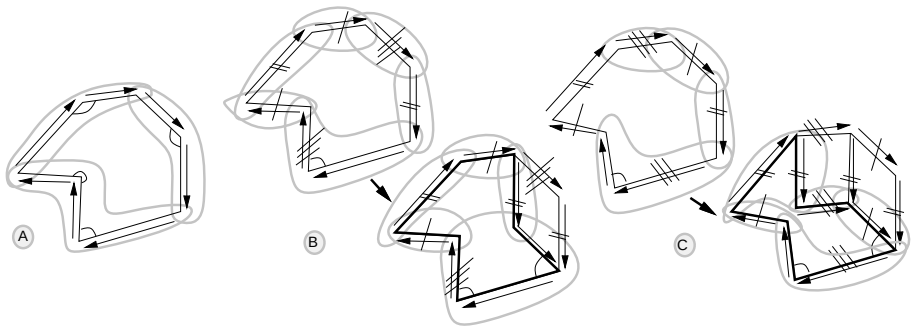

Figure 10: $n-3$ angles and n distances

- All the segments belonging to the same CA set are adjacent (*cf.* Figure 10A). Thus, three CD sets cover the n points and rule **T1** leads to the result.
- The CA set is disconnected or several CA sets are mixed together. Applications of the parallelogram rule transform this case into the previous one (for example, case B and C of Figure 10).

4.2 Nearly simple models

It has been seen that the simple models can be computed by the method. This class of models is large, but the condition for the segments to belong to a cycle is still very restraining. Many practical examples, that can also be solved by the method, do not satisfy this condition (*cf.* Figure 11). Therefore we are now going to describe a larger class of models having the same properties with regard to the algorithm. The elements of this class are called "nearly simple" models. Intuitively, nearly simple models are

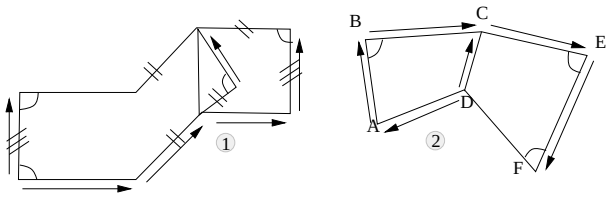

Figure 11: Nearly simple models

models that can be "decomposed" into simple models or treated sequentially by our algorithm as a union of simple models. Definition 2 is more formal

Definition 2 *Let F be a model, i.e. a set of points with distance constraints on some of them and angle constraints on segments joining some of the points. F is said "nearly simple" if there exists a sequence of simple models $F_1, ..., F_n$ such that the following are true:*

1. *F contains F_1*
2. *$\bigcup_{i \leq n} N_i = N$ where $G_{F_i} = (N_i, E_i)$ and $G_F = (N, E)$*
3. *let $(C_i)_{i \leq n}$ and C be the sets of constraints of respectively $(F_i)_{i \leq n}$ and F. Then,*
   (a) *$C = \bigcup_{i \leq n}(C_i \cap C)$*
   (b) *if $i > 1$, C_i can be decomposed in $C_i = (C \cap C_i) \cup D_i \cup A_i$ where*
      - *D_i is composed of distance constraints on couple of points such that there exists N_j with $j \leq i$ containing them,*
      - *A_i is composed of angle constraints on couple of segments joining points such that there exists N_j with $j \leq i$ containing them.*

Then, using the decomposition in simple models and proposition 1, Proposition 2 can be stated:

Proposition 2 *If the model is nearly simple, completely constrained and if the numerical values are consistent, the algorithm computes an evaluated version that satisfies the constraints.*

Note that methods using only one frame of reference in the computation as Brüderlin, Aldefeld and Sunde's first method [14, 15, 16] cannot solve all the simple and the nearly simple models: for example, to solve case 2 of Figure 11, two frames are needed, one where A, B and C are joined and another to join C, E and F. Hence, using several frames during the computation enlarge the set of models resolvable by a rule based approach. Moreover, as the parallelogram rule is not mentioned in the previous models, we cannot ensure whether Sunde's last method [17] computes nearly simple models.

4.3 Models that are not nearly simple

When a model is not nearly simple, it contains a subset composed of n points and segments with $2n - 3$ distance and/or angle constraints on the segments that cannot be decomposed into simple models. Typically such type of subsets are special cases, where the n unknowns are involved simultaneously. The value of n is not bounded. In fact, Proposition 3 can be stated:

Proposition 3 *For any value of n, there exists an m, with $m > n$, and a model F constraining m points with $2m - 3$ angle and distance constraints such that F does not contain a simple model.*

We will prove this proposition showing a way to build a family of models F_1, F_2,... such that:

$$\forall i, F_i \text{ constrains } 3 + i \text{ points and } F_i \text{ does not contain a simple model.}$$

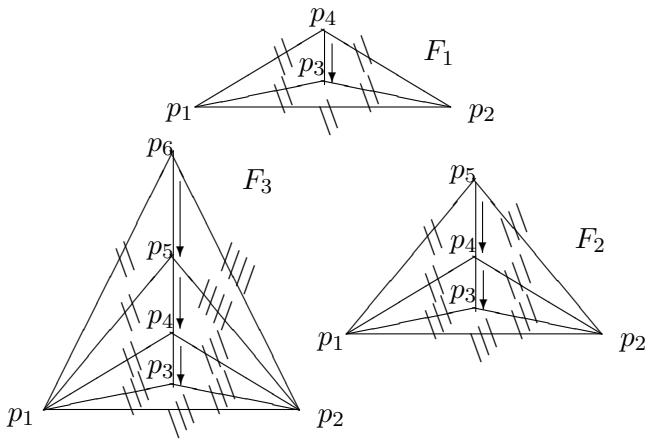

Figure 12: F_1, F_2 and F_3

We will exhibit F_1 and give the way to build F_{i+1} from F_i when $i > 1$ (F_1, F_2 and F_3 are represented in Figure 12):

- F_1 has a distance constraint on p_3p_4 and has the constraint angles (p_1p_3, p_1p_4), (p_1p_3, p_1p_2), (p_1p_2, p_2p_3) and (p_2p_3, p_2p_4).
- For $i > 1$, F_{i+1} contains $j = 3 + i$ points and has the constraints of F_i minus
  - the angle constraint (p_2p_{j-1}, p_2p_j) if j is even or
  - the angle constraint (p_1p_{j-1}, p_1p_j) if j is odd

  and plus the three constraints involving the new point :
  - a distance constraint on $p_{j-1}p_j$ and
  - two angles constraints (p_1p_{j-1}, p_1p_j) and (p_2p_{j-1}, p_2p_j).

  The fact that F_i does not contain a simple model is obvious by induction using this construction.

To compute these models numerically, one can use the fact that the area of the upper triangle is equal to the sum of the areas of the little ones. This equation involves all the unknowns of the model.

Thus, the rule-based methods have their intrinsic limitations: a finite set of rules cannot solve all the 2D models constrained by angles and distances. In fact, if the constraints are seen as equations, the rule-based methods are a way to decompose the computation of the set of non linear equations into a sequence of computations of subsystems of bounded size. This decomposition is impossible when the set must be solved globally. Nevertheless, this method solves the typical models of mechanical drawings and can be used in this context.

5 Implementation

5.1 Characteristics of the expert system

An existing expert system shell that was developed by Benoît Faller (*cf.* [20]) was used. This was written in C and it has the following characteristics:

- If several rules are applicable at the same time, it manages the triggering of the rules with respect to the priority order given by the user,
- It allows actions in the rules, which are calls to procedures written in C,
- It allows variables in the conditions.

Each rule is structured as follows :

- The name of the rule , i.e. R<word or number> and a priority order for the triggering of the rule.
- A list of conditions for the facts of the system's base of facts. These conditions may contain variables (?x means that x is a variable) and they can be negated. In this case, it means that the rules can be triggerered off when no instantiation of the fact appears in the base of facts. Comparisons of the values of variables are allowed conditions.
- Action in the rule, i.e. call of a procedure which will give a list of new facts as a result. These procedures are used here to compute the positions of the points, the angles between two CA sets, etc... They return facts that are inserted in the base of facts.
- Insertion or deletion of facts.

Example:

```
RuleS1 priorite 0 si        (orientation ?seg1 ?ca1 ?alpha)
                            (orientation ?seg1 ?ca2 ?beta)
                            -(equivca ?ca1 ?ca2 ?delta)
                et si (<> ?ca1 ?ca2)
                essayer (directe ?ca1 ?ca2 ?alpha ?beta)
                alors -(orientation ?seg1 ?ca2 ?beta)*
```

If the base of facts contains:

- (orientation s1 ca2 0.)
- (orientation s1 ca1 90.)
- (orientation s2 ca2 180.)

as ca1 and ca2 are different, the rule RuleS1 is applicable.

Then the procedure directe(ca1,ca2,90.,0.) will be called and the fact (orientation s1 s2 0.) will be deleted.

The relative order for the triggering of the rules is the following:

1. the verification rules, to detect overdeterminations as soon as a new constraint is inserted,
2. the "cleaning" rules, to delete facts after the union of CA or CD sets,
3. the "intermediate" rules, like the rules **S1** and **S2**
4. the triangle rules **T1**, **T2**, **T3** and the quadrilateral rule **Q**,
5. the parallelogram rule.

5.2 Facts used in rules

The facts used in the rules can be classified into three families

1. entrance facts
2. facts involving CA sets
3. facts involving CD sets

These facts are described further below.

5.2.1 Entrance facts

These facts are generated when the user inserts a new constraint or enters the geometry of the model. Thus, there is a type of fact corresponding to each type of constraint. As we deal with segments and points, another type of fact denotes the topology of the model. Thus, there are the following facts:

- **(adjacent p seg)** the point p is one of the extremities of the segment seg.
- **(original p x y)** x and y are the coordinates of the point p in the originally entered model.
- **(distance p1 p2 d)** the distance between p1 and p2 is equal to d.
- **(angle seg1 seg2 alpha)** the angle taken in counterclockwise direction between the directed segments seg1 and seg2 is equal to alpha.

5.2.2 Facts involving CA sets

All the facts involving CA sets denote the angular position of objects with respect to a CA set:

- **(orientation seg ca alpha)** for a segment seg
- **(angle_ca r1 ca alpha)** for a CD set r1
- **(equivca ca1 ca2 alpha)** for two CA sets ca1 and ca2

5.2.3 Facts involving CD sets

A frame of reference is associated with each CD set. In these facts, the same name is used for the CD set and for its frame of reference.

- **(position p r1 a b)** the point p, element of the CD set r1, has the coordinates (a, b) in r1.
- **(on_line p1 r1 A B C p2)** the point p1 belongs to the line satisfying the equation $Ax + By + C = 0$ in the CD set r1 and passing through the point p2 whose position is known in r1.
- **(rotation r1 r2 a1 b1 a2 b2 p)** the CD sets r1 and r2 have the point p in common. Its coordinates are (a1,b1) in r1 and (a2,b2) in r2.
- **(transl_R r1 r2 a1 b1 A B C alpha p1 p2)** the CD sets r1 and r2 are fixed in direction by the angle alpha which is the angle between r1 and r2. r2 is free to move on a line satisfying the equation $Ax + By + C = 0$ in r1 and passing through the point p2, known in r1. More precisely, the point p1 belongs to this line and (a1, b1) are its coordinates in r2.
- **(transl_C r1 r2 r3 a1 b1 a2 b2 a3 b3 a4 b4 alpha p1 p2)** the CD sets r1 and r2 are fixed in direction by the angle alpha which is the angle between r1 and r2. r2 is free to move on a circle whose center is p1, known in r1, and whose radius is the distance between p1 and p2. More precisely, p1 has the coordinates (a1, b1) in r1 and (a2, b2) in r3 and p2 has the coordinates (a3, b3) in r2 and (a4, b4) in r3.
- **(equiv r1 r2 m11 m12 m13 m23)** the two CD sets are fixed in translation and in rotation and the transformation from r2 to r1 has the matrix :

$$\begin{pmatrix} m11 & m12 & m13 \\ -m12 & m11 & m23 \\ 0 & 0 & 1 \end{pmatrix}$$

6 Example of working of system

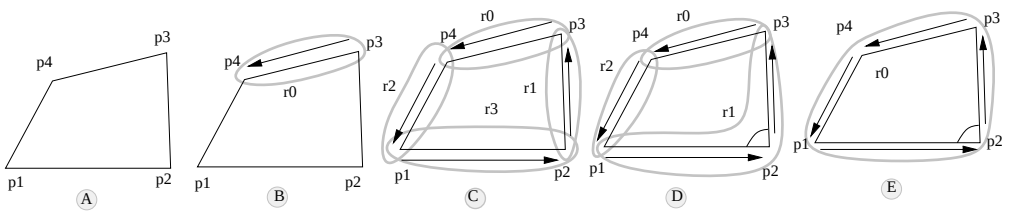

Figure 13: Example

The following example is shown in Figure 13. The user's insertion of facts is preceded with a ← and the insertions caused by the rules are preceded by a ⇒. The "-" preceding the fact denotes the deletion of the fact from the base of facts.

The user first enters the topology of the model and the coordinates of the points in the original model (case A of Figure 13):

← (original p2 306 -259)
← (original p1 143 -255)
← (original p4 119 -209)
← (original p3 265 -141)
← (adjacent p4 p3p4).....

The user inserts a distance constraint. A CD set r0 associated with the CA set ca0 is created, as illustrated in case B of Figure 13 :

← (distance p3 p4 161.000)
⇒ (position p3 r0 0.00 0.00)
⇒ (position p4 r0 161.000 0.00)
⇒ (angle_ca r0 ca0 0.00)
⇒ -(distance p3 p4 161.000)
⇒ (orientation p3p4 ca0 0.000)

The insertion of three distance constraints follows (*cf.* case C of Figure 13):

← (distance p2 p3 106.000)
⇒ (position p2 r1 0.00 0.00)
⇒ (position p3 r1 106.000 0.00)
⇒ (angle_ca r1 ca1 0.00)
⇒ (orientation p2p3 ca1 0.000)
⇒ -(distance p2 p3 106.000)
⇒ (rotation r0 r1 0.00 0.00 106.00 0.00 p3)
← (distance p1 p4 51.000)
⇒ (position p1 r2 0.00 0.00)
⇒ (position p4 r2 51.000 0.00)
⇒ (angle_ca r2 ca2 0.00)
⇒ (orientation p1p4 ca2 0.000)
⇒ -(distance p1 p4 51.000)
⇒ (rotation r0 r2 161.00 0.00 51.00 0.00 p4)
← (distance p1 p2 163.000)
⇒ (position p1 r3 0.00 0.00)
⇒ (position p2 r3 163.000 0.00)
⇒ (angle_ca r3 ca3 0.00)
⇒ (orientation p1p2 ca3 0.000)
⇒ -(distance p1 p2 163.000)
⇒ (rotation r2 r3 0.00 0.00 0.00 0.00 p1)
⇒ (rotation r1 r3 0.00 0.00 163.00 0.00 p2)

Then the user inserts an angle constraint which induces the triggering of rule S1:

← (angle p1p2 p2p3 -249)
⇒ (orientation p1p2 ca1 111.000)
⇒ -(angle p1p2 p2p3 -249)
⇒ (equivca ca1 ca3 -111.000)
⇒ -(orientation p1p2 ca3 0.000)

The two CD sets r2 and r3 have the point p2 in common and are fixed in direction. Rule S2 computes their union (*cf.* case D of Figure 13):

⇒ (equiv r1 r3 -0.358368 0.933580 58.414 152.174)
⇒ -(position p2 r3 163.00 0.00)
⇒ (position p1 r1 58.410 152.170)
⇒ -(position p1 r3 0.00 0.00)
⇒ -(rotation r1 r3 0.00 0.00 163.00 0.00 p2)
⇒ (rotation r1 r2 58.410 152.170 0.00 0.00 p1)
⇒ -(rotation r2 r3 0.00 0.00 0.00 0.00 p1)

The CD sets r0, r1 and r3 are in rotation and rule **T1** is actioned:
⇒ (equiv r0 r2 0.188861 0.982004 151.36807 50.08219)
⇒ -(position p3 r1 106.00 0.00)
⇒ -(position p2 r1 0.00 0.00)
⇒ (position p2 r0 -1.740 105.990)
⇒ -(position p1 r1 58.410 152.170)
⇒ (position p1 r0 151.598 50.623)

All the points belong to the CD set r0 (case H of Figure 13). The model satisfying the constraints is displayed and the session is finished.
This model is computed in 0.9 second by the authors' system on a Sparc Station 1+. Note the following:

- The computing time depends on the size of the model and also on the number of CD or CA sets used in the resolution. For example, the CPU time needed for models involving 8 points can vary from 1.5 to 2.2 seconds.
- The computation time can differ for the same model when the insertion order of the constraints is changed, but the resulting figure is the same.
- When the model is under-constrained, the figure is partially fixed. The user can only have the list of the CD and CA sets present in the base of facts and the position of the points and the angles of the segments w.r.t. these sets. Numerical methods could be used to compute a solution using these sets instead of using the set of equations induced by the constraints.

7 Conclusion

It has been shown that using rules to compute parameterized mechanical designs is feasible. The set of models the proposed system can solve is sufficiently general and essentially comprises the models that are of relevance to 2D mechanical designs.

Experiments with an implementation of this approach have indicated that the computation of constrained models is fast.

The amount of memory used during the session is not negligible. As the rules contain variables, the system maintains the possible instantiations of the premises of the rules during all the session to accelerate the deduction process. An improvement would be to structure the drawings in order to reduce the number of variables during the execution process.

The scope of the method has been studied and the intrinsic limitations of rule oriented approaches have been shown. An idea for improving the resolution would be to mix numerical and rule oriented methods.

A detailed description of this approach and an extension to 3D models can be found in [19].

Acknowledgments

The work described in this paper has been sponsored by Hewlett Packard GmbH. The authors would like to thank Prof. Claude Puech, head of the Graphics group at LIENS and Dr. Steve Hull from Hewlett-Packard GmbH for valuable discussions about various aspects of this research project.

References

[1] D. Roller, F. Schonek, and A. Verroust. Dimension-driven geometry in CAD : A survey. In *Theory and Practice of Geometric Modeling*, pages 509–523. Springer Verlag, 1989.

[2] F. Fitzgerald. Using axial dimensions to determine the proportions of line drawings in computer graphics. *Computer Aided Design*, 13(6):377–382, November 1981.

[3] D. Gossard, R. Zuffante, and H. Sakurai. Representing dimensions, tolerances and features in MCAE systems. *IEEE Computer Graphics and Applications*, 8:51–59, March 1988.

[4] U. Cugini, C. Devoti, and P. Galli. System for parametric definition of engineering drawings. In *MICAD'85*, 1985.

[5] R. Hillyard and I. Braid. Analysis of dimensions and tolerances in computer-aided mechanical design. *Computer Aided Design*, 10(3):161–166, May 1978.

[6] R. Hillyard and I. Braid. Characterizing non ideal shapes in terms of dimensions and tolerances. *Computer Graphics (SIGGRAPH'78)*, 12(3):234–238, August 1978.

[7] R. Light. Symbolic dimensioning in computer-aided design. Master's thesis, MIT, May 1979.

[8] R. Light, V. Lin, and D. Gossard. Variational geometry in CAD. *Computer Graphics (SIGGRAPH'81)*, 15(3):171–177, August 1981.

[9] R. Light and D. Gossard. Modification of geometric models through variational geometry. *Computer Aided Design*, 14(4):209–214, July 1982.

[10] L. Lee and G. Andrews. Inference of the positions of components in an assembly : Part 2. *Computer Aided Design*, 17(1):20–24, January 1985.

[11] W. Chyz. Constraint management for CSG. Master's thesis, M.I.T., June 1985.

[12] B. Brüderlin. Using Prolog for constructing geometric objects defined by constraints. In *European Conference on Computer Algebra*, pages 448–459, 1985.

[13] G. Sunde. Specification of shape by dimensions and other geometric constraints. In *IFIP WG. 5.2 on Geometric Modeling*, Rensselaerville,NY, May 1986.

[14] B. Brüderlin. Constructing three-dimensional geometric objects defined by constraints. In *1986 Workshop on Interactive 3D Graphics*, pages 111–129, Chapell Hill, NC, October 1986.

[15] B. Aldefeld. Variation of geometries based on a geometric-reasoning method. *Computer Aided Design*, 20(3):117–126, April 1988.

[16] G. Sunde. A CAD system with declarative specification of shape. In *EuroGraphics Workshop on Intelligent CAD Systems*, pages 90–105, Noordwijkerhout, The Nederlands, April 1987.

[17] G. Sunde and V. Kallevik. A dimension-driven CAD system. utilizing AI techniques in CAD. Senter for Industriforskning, report no 860216-1, December 1987.

[18] D. Roller. A system for interactive variation design. In *Geometric modeling for Product Engineering, 1988 IFIP/NSF Workshop on Geometric Modeling*. North Holland, 1989.

[19] A. Verroust. Etude de problèmes liés à la définition, la visualisation et l'animation d'objets complexes en Informatique graphique. Thèse d'Etat, Université PARIS-SUD, Orsay, December 1990.

[20] B. Faller. Le système METRO. rapport préliminaire. Rapport Interne du L.R.I., Faculté d'Orsay, FRANCE, April 1988.